Manual Técnico e Documentação de Projeto

Inclusiva Play

INSTITUTO FEDERAL
Paraná
Campus
Pinhais

Versão: 1.0

Data: 06 de julho de 2025

Autor: Rodrigo de Lima

Orientadores: Lauriana Paludo, Eliana maria dos santos

Controle de Versões

Versão	Data	Autor	Descrição
1.0	05/07/2025	Rodrigo de Lima	Versão inicial da documentação completa do projeto.

Sumário

Capítulo 6: Roadmap de Evolução e Conformidade LGPD

- 6.1. Visão Geral dos Próximos Passos
- 6.2. Plano de Adequação à LGPD (Prioridade)
    - 6.2.1. Criptografia de Dados Sensíveis
    - 6.2.2. Implementação da Exclusão de Dados sob Demanda
    - 6.2.3. Política de Retenção e Exclusão Automática
- 6.3. Implementação da Geolocalização
- 6.4. Trabalhos Futuros

Apêndice A: Código Fonte Completo

Apêndice B: Script do Banco de Dados (SQL)

Capítulo 1: Introdução ao Projeto

1.1. Visão Geral e Propósito

A "Inclusiva Play" é uma plataforma web desenvolvida para ser a principal ponte de conexão entre famílias de crianças com Transtorno do Espectro Autista (TEA) e uma rede de profissionais e centros esportivos especializados em atividades adaptadas. O projeto visa facilitar o acesso a serviços de qualidade, promover a inclusão social, o desenvolvimento motor e o bem-estar de crianças com TEA. A justificativa para sua criação reside no potencial de impacto social e na resposta a uma demanda clara da comunidade, amparada pela Lei nº 12.764/2012 (Lei Berenice Piana), que garante o direito à saúde, educação e lazer para pessoas com TEA.

1.2. Escopo e Objetivos

O objetivo geral do projeto é desenvolver e manter uma aplicação web funcional, segura e intuitiva.

Objetivos Específicos alcançados:

- Implementar um sistema seguro de cadastro e autenticação para "Pais/Responsáveis" e "Profissionais".
- Desenvolver um painel de gestão completo (CRUD) para os profissionais gerenciarem seus perfis.
- Construir uma interface de usuário intuitiva para a busca e visualização de perfis.
- Implementar um sistema de agendamento funcional com visão de calendário.
- Criar um dashboard analítico com métricas da plataforma.
- Desenvolver uma plataforma de conteúdo (blog) para a comunidade.

1.3. Público-Alvo e Personas

1.3.1. Persona: Pai/Mãe (Usuário)

- **Nome:** Ana, 38 anos.
- **Perfil:** Moradora de Pinhais, PR, mãe de Léo, 7 anos, com TEA Nível 2. Trabalha em período integral.
- **Necessidades:** Encontrar profissionais qualificados e locais de atividade próximos, ler avaliações e agendar sessões de forma simples.

1.3.2. Persona: Profissional

- **Nome:** Carlos, 30 anos.
- **Perfil:** Educador Físico autônomo com especialização em psicomotricidade.
- **Necessidades:** Divulgar seu trabalho para um público focado, gerenciar seus horários e construir uma reputação online.

1.3.3. Persona: Beneficiário Final (Adolescente com TEA)

- **Nome:** Eduardo Novick.
- **Perfil:** Adolescente (13-18 anos) de Pinhais, cursando o Ensino Médio e trabalhando como estagiário.
- **Desafios:** Enfrenta barreiras sociais e preconceito, sentindo-se inferiorizado e isolado em ambientes não inclusivos.
- **Propósito da Plataforma para Ele:** A "Inclusiva Play" existe para criar ambientes seguros onde jovens como Eduardo possam se desenvolver, socializar e fortalecer a autoestima através do esporte, combatendo os desafios que ele enfrenta.

Capítulo 2: Guia de Funcionalidades

2.1. Fluxo do Usuário (Pais/Responsáveis)

1. **Acesso e Cadastro:** O usuário acessa a index.php e pode se cadastrar através da cadastro_usuario.php.
2. **Autenticação:** Realiza o login na login_usuario.php. Pode recuperar a senha via esqueceu_senha.php.
3. **Painel Principal (pagina_usuario.php):** Após o login, acessa seu painel, de onde pode navegar para as áreas de atividades (atividades.php), agendamento (agenda_profisional.php), e blog (blog.php).
4. **Agendamento:** Na página de agendamento, pode selecionar um profissional, data e horário para marcar uma consulta

2.2. Fluxo do Profissional

1. **Cadastro e Login:** O profissional se cadastra na cadastro.php e acessa o sistema via login.php.
2. **Painel de Gestão (gestao_profissionais.php):** É a tela principal, onde o profissional pode ver a lista de outros profissionais, buscar, e acessar os demais painéis.
3. **Gerenciamento:** Pode editar seu perfil (editar.php), alterar sua senha e foto (configuracoes.php), ou ser excluído (ação de um admin, excluir.php).
4. **Visualização de Agenda:** Acessa dashboard_agendamentos.php para ver a lista de compromissos ou dashboard_calendario.php para a visão em calendário.
5. **Análise de Dados:** Acessa o dashboard_profissionais.php para visualizar gráficos sobre a plataforma.

2.3. Funcionalidades Públicas

- **Home (index.php):** Apresenta o projeto.
- **Páginas Informativas:** sobre_nos.php, especialidades.php, faq.php, lei_berenici.php.
- **Parcerias (parceiros.php):** Informa sobre como apoiar o projeto.

Capítulo 3: Arquitetura do Sistema

3.1. Visão Geral da Arquitetura

A aplicação adota uma arquitetura de servidor tradicional, com uma abordagem híbrida de programação:

- **Procedural:** A maior parte dos scripts de front-end com PHP embutido segue um fluxo procedural.
- **Orientada a Objetos (POO):** Uma estrutura MVC (Model-View-Controller) foi implementada para a API de gestão de profissionais, separando a lógica de negócio (Model), o controle de requisições (Controller) e a resposta (View, no caso, JSON)

3.2. Detalhamento do Backend (PHP)

- **Conexão com Banco de Dados:** Utiliza dois métodos, PDO (database.php) e MySQLi (conexao.php), ambos com o uso de *prepared statements* para segurança.
- **Gerenciamento de Sessão:** session_start() é usado no topo dos arquivos protegidos para controlar o acesso de usuários logados (gestao_profissionais.php, pagina_usuario.php, etc.).
- **Lógica de Negócio:** Os arquivos ProfessionalModel.php e ProfessionalController.php centralizam a lógica de manipulação dos dados dos profissionais.

3.3. Detalhamento do Frontend (HTML/CSS/JS)

- **Estrutura:** As páginas são construídas com HTML5 semântico.
- **Estilização:** CSS3 puro é utilizado, com bom uso de variáveis (:root) para manter a consistência visual (parceiros.php, gestao_profissionais.php).
- **Interatividade:** JavaScript puro (Vanilla JS) é usado para manipulação do DOM, como a lógica de filtros em especialidades.php, os modais em index.php, e a lógica de exibição condicional em cadastro_parceiros.php.

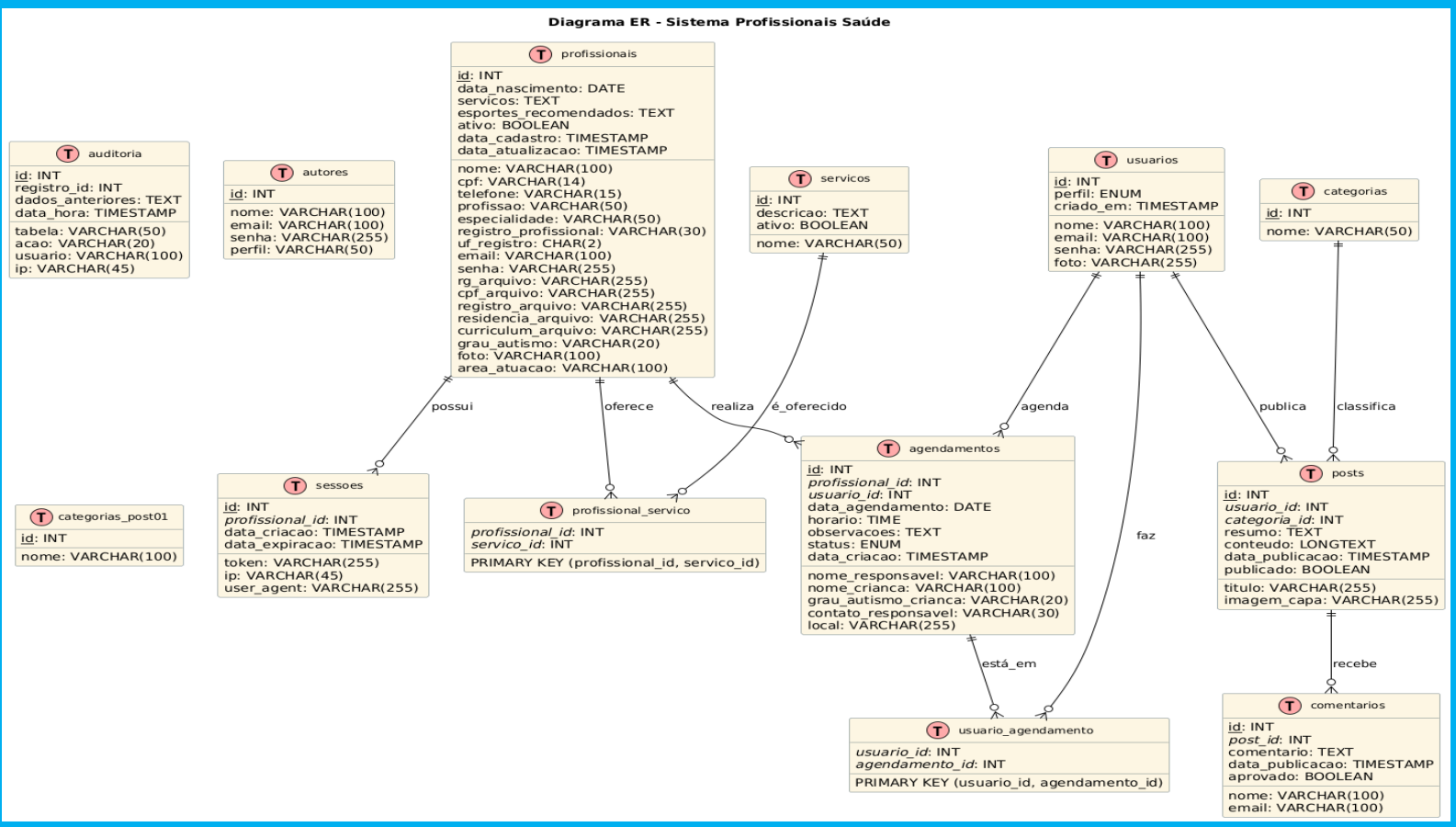
3.4. Bibliotecas e Dependências Externas

- **Chart.js:** Para gráficos no dashboard (dashboard_profissionais.php).
- **FullCalendar:** Para o calendário interativo de agendamentos (dashboard_calendario.php).
- **Leaflet.js & Mapillary-JS:** Para a prototipagem do mapa (mapa_2.0_teste.php).
- **Font Awesome:** Para a iconografia do sistema.
- **Animate.css:** Para animações na página inicial.

Capítulo 4: Banco de Dados

4.1. Modelo Entidade-Relacionamento (DER)

O DER abaixo ilustra a estrutura completa do banco de dados, mostrando todas as entidades e seus relacionamentos.



4.2. Descrição das Tabelas Principais

Tabela	Descrição
profissionais	Armazena os dados cadastrais, de contato e de autenticação dos profissionais.
usuarios	Armazena os dados dos pais/responsáveis que utilizam a plataforma.
agendamentos	Registra os detalhes de cada agendamento, como data, hora, e qual profissional foi contratado.
usuario_agendamento	Tabela de ligação que associa um usuario_id a um agendamento_id.
posts	Contém o conteúdo dos artigos do blog, como título, resumo e corpo do texto.
comentarios	Armazena os comentários feitos nos posts, aguardando aprovação.
auditoria	Tabela de log que registra automaticamente as alterações feitas na tabela profissionais.

4.3. Views e Triggers

- **View vw_profissionais:** Foi criada para simplificar a consulta de profissionais com seus serviços associados, otimizando a performance.
- **Triggers de Auditoria:** Foram implementados triggers de INSERT, UPDATE e DELETE na tabela profissionais para registrar todas as modificações na tabela auditoria, garantindo a rastreabilidade dos dados.

Capítulo 5: Guia de Instalação e Configuração

5.1. Pré-requisitos

- Servidor web local WAMP, com suporte a PHP 7.4 ou superior.
- Servidor de banco de dados MySQL.
- Um cliente de banco de dados (phpMyAdmin).

5.2. Passos para Instalação Local

1. **Código-Fonte:** Coloque todos os arquivos do projeto na pasta raiz do seu servidor local (ex: htdocs no XAMPP).
2. **Banco de Dados:** Usando um cliente de sua preferência, crie um novo banco de dados chamado sistema_profissionais_saude.
3. **Importação:** Execute o script SQL completo para criar todas as tabelas, views e triggers.
4. **Configuração:** Abra o arquivo config.php e/ou database.php e verifique se as credenciais do banco de dados (DB_HOST, DB_USER, DB_PASS, DB_NAME) correspondem à sua configuração local.
5. **Acesso:** Abra seu navegador e acesse http://localhost/[nome_da_pasta_do_projeto]/.

Capítulo 6: Roadmap de Evolução e Conformidade LGPD

6.1. Visão Geral dos Próximos Passos

A base da plataforma "Inclusiva Play" está sólida e funcional. Os próximos passos focam em finalizar funcionalidades críticas, aprimorar a segurança e conformidade, e adicionar novos módulos de valor.

6.2. Plano de Adequação à LGPD (Prioridade)

A conformidade com a Lei Geral de Proteção de Dados é uma prioridade máxima. O plano de ação é:

6.2.1. Criptografia de Dados Sensíveis em Repouso

- **O quê:** Dados como caminhos de arquivos de documentos (rg_arquivo, cpf_arquivo) e outras informações pessoais sensíveis no banco de dados devem ser criptografados.
- **Como:** Utilizar as funções openssl_encrypt() e openssl_decrypt() do PHP com uma chave de criptografia segura armazenada em uma variável de ambiente (fora do código-fonte) para garantir que, mesmo com acesso ao banco, os dados não possam ser lidos.

6.2.2. Implementação da Exclusão de Dados sob Demanda

- **O quê:** Criar uma funcionalidade na área de perfil do usuário (pagina_usuario.php ou configuracoes.php) que permita solicitar a exclusão de sua conta e todos os dados associados.
- **Como:**
    1. Adicionar um botão "Excluir minha conta" na interface.
    2. Ao clicar, exibir um modal de confirmação, informando que a ação é irreversível.
    3. Se confirmado, acionar um script PHP (excluir_minha_conta.php) que executará uma transação SQL para apagar os dados do usuário das tabelas usuarios, agendamentos, posts, etc., e também removerá os arquivos de upload associados (fotos) do servidor.

6.2.3. Política de Retenção e Exclusão Automática

- **O quê:** Definir uma política para exclusão automática de dados de contas inativas para minimizar a retenção de dados desnecessários.
- **Como:**
    1. **Definir a regra:** Por exemplo, "Contas de usuários que não realizam login por um período de 2 anos serão marcadas para exclusão."
    2. **Implementar um Script:** Criar um script PHP (cron_limpeza_lgpd.php) que busca por usuários inativos de acordo com a regra.
    3. **Automatizar:** Configurar um **Cron Job** no servidor para executar este script periodicamente (ex: uma vez por mês) para realizar a limpeza de forma automática.

6.3. Implementação da Funcionalidade de Geolocalização

- Finalizar a integração da API de mapas. A recomendação é aprofundar o uso do **Leaflet.js** (já prototipado em mapa_2.0_teste.php) com um provedor de geocodificação como o **Nominatim (OpenStreetMap)** para evitar os custos e restrições da API do Google Maps. A funcionalidade deve conectar-se em tempo real ao banco de dados para plotar os profissionais.

6.4. Trabalhos Futuros

- **Sistema de Avaliação:** Implementar a funcionalidade para que usuários possam avaliar e comentar sobre os profissionais.
- **Cadastro de Centros Esportivos:** Criar a entidade e a interface para o cadastro de academias e clubes.
- **Agenda de Eventos:** Adicionar um módulo para a divulgação de eventos inclusivos.
- **Refatoração:** Refatorar progressivamente o código para um padrão MVC mais consistente, melhorando a organização e a manutenibilidade a longo prazo.

Apêndice A: Código Fonte Completo

Nesta seção, seriam inseridos os conteúdos completos de todos os 65 arquivos de código que compõem o projeto. Para fins de organização, os arquivos estão listados abaixo.

Arquivos de Código (.php, .html) (65)

1. admin_comentarios.php
2. agenda_profisional.php
3. alterar_senha.php
4. api.php
5. atividades.php
6. atualziar_dados.php
7. atualziar_foto.php
8. beneficio-certificado.html
9. beneficio-conteudo.html
10. beneficio-eventos.html
11. beneficio-fiscal.html
12. beneficio-networking.html
13. beneficio-relatorios.html
14. beneficio-visibilidade.html
15. blog.php
16. blog_home.php
17. cadastro.php
18. cadastro_parceiros.php
19. cadastro_usuario.php
20. cancelar_agendamento.php

21. categorias.php
22. categorias_editar.php
23. categorias_excluir.php
24. config.php
25. configuracoes.php
26. conexao.php
27. contato.php
28. dashboard_agendamentos.php
29. dashboard_calendario.php
30. dashboard_profissionais.php
31. database.php
32. editar.php
33. especialidades.php
34. esqueceu_senha.php
35. excluir.php
36. excluir_agendamento.php
37. faq.php
38. gestao_profissionais.php
39. index.php
40. lei_berenici.php
41. listar_profissionais.php
42. localização.php
43. login.php

44. login_usuario.php
45. logout.php
46. logout_usuario.php
47. mapa_2.0_teste.php
48. novo_post.php
49. pagina_uso.php
50. pagina_usuario.php
51. parceiros.php
52. perfil.php
53. politica-privacidade.php
54. pot.php
55. professionalcontroller.php
56. professionalMOdel.php
57. recuperar_senha.php
58. redefinir_senha.php
59. remover_foto.php
60. remover_foto_usuario.php
61. salvar_profissional.php
62. sobre_nos.php
63. termos-parceira.php
64. trocar_foto_usuario.php
65. ver_post.php

Apêndice B: Script do Banco de Dados (SQL)

Nesta seção, seria inserido o script SQL completo para a criação do banco de dados e de todas as suas estruturas. Abaixo, um trecho exemplificativo.

SQL

```
-- ULTIMO BANCO ATUALZIADO EM  - 01 / 07 / 2025
-- TUDO FUNCIONANDO - CORRETAMENTE  , NESTE AQUI - PELO MENOS OS PRINCIPAIS


-- Criação do banco de dados
CREATE DATABASE IF NOT EXISTS sistema_profissionais_saude
CHARACTER SET utf8mb4
COLLATE utf8mb4_unicode_ci;
```

```
USE sistema_profissionais_saude;

-- Tabela de profissionais (tabela principal)
CREATE TABLE IF NOT EXISTS profissionais (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(100) NOT NULL,
    cpf VARCHAR(14) NOT NULL UNIQUE,
    data_nascimento DATE NOT NULL,
    telefone VARCHAR(15) NOT NULL,
    profissao VARCHAR(50) NOT NULL,
    especialidade VARCHAR(50),
    registro_profissional VARCHAR(30) NOT NULL,
    uf_registro CHAR(2) NOT NULL,
    servicos TEXT,
    email VARCHAR(100) NOT NULL UNIQUE,
    senha VARCHAR(255) NOT NULL,
    rg_arquivo VARCHAR(255),
    cpf_arquivo VARCHAR(255),
    registro_arquivo VARCHAR(255),
    residencia_arquivo VARCHAR(255),
    curriculum_arquivo VARCHAR(255),
    ativo BOOLEAN DEFAULT TRUE,
    data_cadastro TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    data_atualizacao TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON

UPDATE CURRENT_TIMESTAMP,
    INDEX idx_profissao (profissao),
    INDEX idx_especialidade (especialidade),
    INDEX idx_uf_registro (uf_registro),
    INDEX idx_email (email)
) ENGINE=InnoDB;
```

```
-- Tabela de auditoria (para registrar alterações importantes)
CREATE TABLE IF NOT EXISTS auditoria (
    id INT AUTO_INCREMENT PRIMARY KEY,
    tabela VARCHAR(50) NOT NULL,
    acao VARCHAR(20) NOT NULL COMMENT 'INSERT, UPDATE, DELETE',
    registro_id INT NOT NULL,
    dados_anteriores TEXT,
    usuario VARCHAR(100),
    ip VARCHAR(45),
    data_hora TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    INDEX idx_tabela (tabela),
    INDEX idx_acao (acao),
    INDEX idx_registro_id (registro_id)
) ENGINE=InnoDB;

-- Tabela de sessões (para controle de login)
CREATE TABLE IF NOT EXISTS sessoes (
    id INT AUTO_INCREMENT PRIMARY KEY,
    profissional_id INT NOT NULL,
    token VARCHAR(255) NOT NULL,
    data_criacao TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    data_expiracao TIMESTAMP NOT NULL,
    ip VARCHAR(45),
    user_agent VARCHAR(255),
    FOREIGN KEY (profissional_id) REFERENCES profissionais(id) ON DELETE
CASCADE,
    INDEX idx_token (token),
    INDEX idx_profissional_id (profissional_id)
) ENGINE=InnoDB;

-- Tabela de serviços (para normalização)
CREATE TABLE IF NOT EXISTS servicos (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(50) NOT NULL UNIQUE,
    descricao TEXT,
    ativo BOOLEAN DEFAULT TRUE
) ENGINE=InnoDB;
```

```
-- Tabela de relação profissional_servico (many-to-many)
CREATE TABLE IF NOT EXISTS profissional_servico (
    profissional_id INT NOT NULL,
    servico_id INT NOT NULL,
    PRIMARY KEY (profissional_id, servico_id),
    FOREIGN KEY (profissional_id) REFERENCES profissionais(id) ON DELETE
CASCADE,
    FOREIGN KEY (servico_id) REFERENCES servicos(id) ON DELETE CASCADE
) ENGINE=InnoDB;

-- Inserção de serviços padrão
INSERT INTO servicos (nome, descricao) VALUES
('Consulta', 'Consulta médica ou de outro profissional'),
('Exame', 'Realização de exames diversos'),
('Procedimento', 'Procedimentos clínicos ou terapêuticos'),
('Orientação', 'Orientação e aconselhamento'),
('Terapia', 'Sessões de terapia'),
('Acompanhamento', 'Acompanhamento contínuo do paciente'),
('Atividade Física Adaptada', 'Exercícios físicos adaptados para necessidades especiais'),
('Psicopedagogia', 'Acompanhamento psicopedagógico'),
('Fonoaudiologia', 'Terapia fonoaudiológica'),
('Terapia Ocupacional', 'Terapia ocupacional'),
('Esporte e Lazer Inclusivo', 'Atividades esportivas e de lazer inclusivas');

-- Trigger para auditoria de profissionais (INSERT)
DELIMITER //
CREATE TRIGGER after_profissionais_insert
AFTER INSERT ON profissionais
FOR EACH ROW
BEGIN
    INSERT INTO auditoria (tabela, acao, registro_id, usuario, ip)
    VALUES ('profissionais', 'INSERT', NEW.id, CURRENT_USER(),
CONNECTION_ID());
END//
DELIMITER ;
```

```
-- Trigger para auditoria de profissionais (UPDATE)
DELIMITER //
CREATE TRIGGER before_profissionais_update
BEFORE UPDATE ON profissionais
FOR EACH ROW
BEGIN
    DECLARE dados_anteriores TEXT;

    SET dados_anteriores = CONCAT(
        'nome=', OLD.nome, ';',
        'email=', OLD.email, ';',
        'profissao=', OLD.profissao, ';',
        'especialidade=', IFNULL(OLD.especialidade, 'NULL'), ';',
        'registro_profissional=', OLD.registro_profissional, ';',
        'ativo=', OLD.ativo
    );

    INSERT INTO auditoria (tabela, acao, registro_id, dados_anteriores, usuario, ip)
    VALUES ('profissionais', 'UPDATE', OLD.id, dados_anteriores,
CURRENT_USER(), CONNECTION_ID());
END//
DELIMITER ;

-- Trigger para auditoria de profissionais (DELETE)
DELIMITER //
CREATE TRIGGER before_profissionais_delete
BEFORE DELETE ON profissionais
FOR EACH ROW
BEGIN
    DECLARE dados_anteriores TEXT;

    SET dados_anteriores = CONCAT(
        'nome=', OLD.nome, ';',
        'email=', OLD.email, ';',
        'profissao=', OLD.profissao, ';',
        'especialidade=', IFNULL(OLD.especialidade, 'NULL'), ';',
        'registro_profissional=', OLD.registro_profissional
    );
```

```
    INSERT INTO auditoria (tabela, acao, registro_id, dados_anteriores, usuario, ip)
    VALUES ('profissionais', 'DELETE', OLD.id, dados_anteriores,
CURRENT_USER(), CONNECTION_ID());
END//
DELIMITER ;

-- View para listagem simplificada de profissionais
CREATE VIEW vw_profissionais AS
SELECT
    p.id,
    p.nome,
    p.profissao,
    p.especialidade,
    p.registro_profissional,
    p.uf_registro,
    p.email,
    p.ativo,
    GROUP_CONCAT(s.nome SEPARATOR ', ') AS servicos_oferecidos,
    p.data_cadastro,
    p.data_atualizacao
FROM
    profissionais p
LEFT JOIN
    profissional_servico ps ON p.id = ps.profissional_id
LEFT JOIN
    servicos s ON ps.servico_id = s.id
GROUP BY
    p.id;
```

```
SHOW DATABASES LIKE 'sistema_profissionais_saude';



USE sistema_profissionais_saude;
SHOW TABLES;

SELECT cpf, email FROM profissionais WHERE email = 'email_inserido_no_login';
SELECT cpf FROM profissionais LIMIT 5;



ALTER TABLE profissionais
ADD COLUMN grau_autismo VARCHAR(20) COMMENT 'Leve/Moderado/Severo',
ADD COLUMN esportes_recomendados TEXT;

-- adicionada esta linha para armazenar foto da pessoa

ALTER TABLE profissionais ADD COLUMN foto VARCHAR(100) DEFAULT
NULL COMMENT 'Caminho da foto do profissional';


select * from profissionais


CREATE TABLE IF NOT EXISTS agendamentos (
    id INT AUTO_INCREMENT PRIMARY KEY,
    profissional_id INT NOT NULL,
    nome_responsavel VARCHAR(100) NOT NULL,
    nome_crianca VARCHAR(100),
    grau_autismo_crianca VARCHAR(20),
    contato_responsavel VARCHAR(30),
    data_agendamento DATE NOT NULL,
    horario TIME NOT NULL,
    observacoes TEXT,
    status ENUM('pendente', 'confirmado', 'cancelado') DEFAULT 'pendente',
    data_criacao TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
```

```
    FOREIGN KEY (profissional_id) REFERENCES profissionais(id) ON DELETE
CASCADE,

    INDEX idx_profissional (profissional_id),
    INDEX idx_data_hora (data_agendamento, horario)
) ENGINE=InnoDB;



-- Categorias do blog
CREATE TABLE IF NOT EXISTS categorias (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(50) NOT NULL UNIQUE
);



-- Autores do blog (podem ser os próprios profissionais ou admins)
CREATE TABLE IF NOT EXISTS autores (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(100) NOT NULL,
    email VARCHAR(100) NOT NULL UNIQUE,
    senha VARCHAR(255) NOT NULL,
    perfil VARCHAR(50) DEFAULT 'editor' -- ex: admin, editor, profissional
);
```

```
-- Posts do blog
CREATE TABLE IF NOT EXISTS posts (
    id INT AUTO_INCREMENT PRIMARY KEY,
    autor_id INT NOT NULL,
    categoria_id INT NOT NULL,
    titulo VARCHAR(255) NOT NULL,
    resumo TEXT,
    conteudo LONGTEXT NOT NULL,
    imagem_capa VARCHAR(255),
    data_publicacao TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    publicado BOOLEAN DEFAULT TRUE,
    FOREIGN KEY (autor_id) REFERENCES autores(id) ON DELETE CASCADE,
    FOREIGN KEY (categoria_id) REFERENCES categorias(id) ON DELETE SET
NULL
);

INSERT INTO posts (categoria_id, titulo, resumo, conteudo, data_publicacao,
publicado) VALUES
(
    1,
    'Terapias Eficazes para Crianças Autistas',
    'Conheça as principais terapias que ajudam no desenvolvimento de crianças com
autismo, incluindo ABA, terapia ocupacional e fonoaudiologia.',
    'O autismo é um transtorno do neurodesenvolvimento que afeta a comunicação e o
comportamento. As terapias mais recomendadas incluem:\n\n- ABA (Análise do
Comportamento Aplicada)\n- Terapia Ocupacional\n- Fonoaudiologia\n\nCada terapia
deve ser adaptada às necessidades individuais da criança para melhores resultados.',
    '2025-06-17 10:30:00',
    1
);
```

```
SELECT * FROM autores WHERE id = 1;

CREATE TABLE IF NOT EXISTS comentarios (
    id INT AUTO_INCREMENT PRIMARY KEY,
    post_id INT NOT NULL,
    nome VARCHAR(100) NOT NULL,
    email VARCHAR(100) NOT NULL,
    comentario TEXT NOT NULL,
    data_publicacao TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    aprovado BOOLEAN DEFAULT FALSE,
    FOREIGN KEY (post_id) REFERENCES posts(id) ON DELETE CASCADE
);


CREATE TABLE IF NOT EXISTS usuarios (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(100) NOT NULL,
    email VARCHAR(100) NOT NULL UNIQUE,
    senha VARCHAR(255) NOT NULL,
    perfil ENUM('admin', 'editor') DEFAULT 'editor',
    criado_em TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

CREATE TABLE categorias_post01 (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(100) NOT NULL UNIQUE
);
```

```
SHOW TABLES;

select * from usuarios


ALTER TABLE profissionais ADD COLUMN area_atuacao VARCHAR(100);

UPDATE profissionais
SET cpf = REPLACE(REPLACE(REPLACE(REPLACE(REPLACE(cpf, '.', ''), '-', ''),
'/', ''), ' ', ''), ',', '');

-- Remove coluna autor_id e chave estrangeira antiga (se existir)
ALTER TABLE posts DROP FOREIGN KEY IF EXISTS fk_posts_autor;
ALTER TABLE posts DROP COLUMN autor_id;



-- Adiciona coluna usuario_id com FK para usuarios.id
ALTER TABLE posts ADD COLUMN usuario_id INT NOT NULL;

ALTER TABLE posts
ADD CONSTRAINT fk_posts_usuario FOREIGN KEY (usuario_id) REFERENCES
usuarios(id) ON DELETE CASCADE;



SHOW CREATE TABLE posts;

SELECT CONSTRAINT_NAME
FROM information_schema.KEY_COLUMN_USAGE
WHERE TABLE_SCHEMA = 'sistema_profissionais_saude'
  AND TABLE_NAME = 'posts'
  AND REFERENCED_TABLE_NAME IS NOT NULL;
```

```
ALTER TABLE posts DROP FOREIGN KEY nome_da_fk;
ALTER TABLE posts DROP COLUMN autor_id;


ALTER TABLE posts ADD COLUMN usuario_id INT NOT NULL;

ALTER TABLE posts
ADD CONSTRAINT fk_posts_usuario FOREIGN KEY (usuario_id) REFERENCES
usuarios(id) ON DELETE CASCADE;

INSERT INTO categorias (nome) VALUES
('Dicas para Pais'),
('Terapias e Intervenções'),
('Saúde e Bem-Estar'),
('Atividades Inclusivas'),
('Educação e Desenvolvimento'),
('Direitos e Acessibilidade'),
('Tecnologia Assistiva'),
('Notícias e Atualizações'),
('Depoimentos de Pais e Profissionais');

SHOW COLUMNS FROM usuarios;
```

```
CREATE TABLE IF NOT EXISTS usuario_agendamento (
   usuario_id INT NOT NULL,
   agendamento_id INT NOT NULL,
   PRIMARY KEY (usuario_id, agendamento_id),
   CONSTRAINT fk_usuario
      FOREIGN KEY (usuario_id) REFERENCES usuarios(id) ON DELETE
CASCADE,
   CONSTRAINT fk_agendamento
      FOREIGN KEY (agendamento_id) REFERENCES agendamentos(id) ON
DELETE CASCADE
) ENGINE=InnoDB;

SELECT DATABASE();

SHOW TABLES;

SHOW CREATE TABLE usuarios;

SHOW CREATE TABLE agendamentos;

SHOW TABLE STATUS WHERE Name IN ('usuarios', 'agendamentos');

CHECK TABLE usuarios;

SHOW TABLE STATUS WHERE Name = 'usuarios';

ALTER TABLE usuarios ENGINE=InnoDB;

ALTER TABLE agendamentos ADD COLUMN local VARCHAR(255) NULL;

ALTER TABLE usuarios ADD COLUMN foto VARCHAR(255) DEFAULT NULL
COMMENT 'Caminho da foto do usuário';
```

```
SELECT a.id, p.nome AS profissional_nome, p.especialidade, a.data_agendamento,
a.horario
FROM usuario_agendamento ua
INNER JOIN agendamentos a ON ua.agendamento_id = a.id
INNER JOIN profissionais p ON a.profissional_id = p.id
WHERE ua.usuario_id = 15 AND a.data_agendamento >= CURDATE()
ORDER BY a.data_agendamento ASC, a.horario ASC;
```

```
SELECT * FROM usuario_agendamento;

SELECT * FROM agendamentos WHERE id = 4;

INSERT INTO usuario_agendamento (usuario_id, agendamento_id) VALUES (13,
ID_DO_AGENDAMENTO);
SELECT id, nome_crianca, data_agendamento FROM agendamentos LIMIT 10;
```

```
ALTER TABLE agendamentos ADD COLUMN usuario_id INT NULL;
ALTER TABLE agendamentos
ADD CONSTRAINT fk_agendamento_usuario FOREIGN KEY (usuario_id)
REFERENCES usuarios(id);
```

Sequência

Tabela	Descrição
profissionais	Armazena os dados cadastrais, de contato e de autenticação dos profissionais.
usuarios	Armazena os dados dos pais/responsáveis que utilizam a plataforma.
agendamentos	Registra os detalhes de cada agendamento, como data, hora, e qual profissional foi contratado.
usuario_agendamento	Tabela de ligação que associa um usuario_id a um agendamento_id.
posts	Contém o conteúdo dos artigos do blog, como título, resumo e corpo do texto.
comentarios	Armazena os comentários feitos nos posts, aguardando aprovação.
auditoria	Tabela de log que registra automaticamente as alterações feitas na tabela profissionais.